	Type	Hits	Search Text	DBs	Time Stamp	Comments	Error Definition	Errors
1	BRS	434	S38 with value	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/13 14:06			
2	BRS	9219	pointer with last	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/13 14:06			
3	BRS	7	S39 and S40	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/13 14:06			
4	BRS	1790	probability with magnitude	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/13 14:05			
5	BRS	58	S33 and probab$9	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 18:19			
6	BRS	1	S34 and probab$9	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 18:17			
7	BRS	10	S28 and probab$9	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 18:17			
8	BRS	2	S28 and S29	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:41			
9	BRS	235	S29 same S30	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:39			
10	BRS	96	S32 and future	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:39			
11	BRS	52	pointer with access$3 with future	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:38			
12	BRS	7700	last near4 access$3	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:38			
13	BRS	14458	pointer with access$3	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:38			
14	BRS	940	S29 and S30	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:38			

	Type	Hits	Search Text	DBs	Time Stamp	Comments	Error Definitio n	Errors
15	BRS	7	S26 and probab$9	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	2005/01/12 17:33			
16	BRS	28	(US-20020042807-$ or US-20020049865-$ or US-20020089901-$ or US-20020104077-$ or US-20020108106-$ or US-20020108107-$ or US-20020112227-$ or US-20020165848-$ or US-20030126208-$ or US-20030212805-$).did. or (US-5448706-$ or US-5475754-$ or US-5822787-$ or US-5835743-$ or US-5848274-$ or US-5898422-$ or US-5937329-$ or US-5991871-$ or US-6029000-$ or US-6047362-$ or US-6115782-$ or US-6219787-$ or US-6295645-$ or US-6314436-$ or US-6349312-$ or US-6535942-$).did. or (EP-679028-$).did.	US-PGPUB; USPAT; DERWENT	2005/01/12 17:16			

US005537573A

United States Patent [19]

Ware et al.

[11] Patent Number: 5,537,573

[45] Date f Patent: Jul. 16, 1996

[54] **CACHE SYSTEM AND METHOD FOR PREFETCHING OF DATA**

[75] Inventors: **Frederick A. Ware**, Los Altos Hills; **Michael P. Farmwald**, Portola Valley; **Craig Hampel**; **Karnamadakala Krishnamohan**, both of San Jose, all of Calif.

[73] Assignee: **Rambus, Inc.**, Mountain View, Calif.

[21] Appl. No.: **69,147**

[22] Filed: **May 28, 1993**

[51] **Int. Cl.**[6] **G06F 12/00**

[52] **U.S. Cl.** **395/464**; 395/421.03; 364/DIG. 1; 364/263.1

[58] **Field of Search** 395/400, 425, 395/250, 444, 445, 464, 421.03

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,807,110	2/1989	Pomerene et al.	395/425
4,943,908	7/1990	Emma et al.	395/425
4,980,823	12/1990	Liu	395/425
5,136,697	8/1992	Johnson	395/375
5,305,389	4/1994	Palmer	395/425

FOREIGN PATENT DOCUMENTS

0157175	10/1985	European Pat. Off.	G06F 12/08
0242595	10/1987	European Pat. Off.	G06F 11/10
0296430	12/1988	European Pat. Off.	G06F 12/08

OTHER PUBLICATIONS

IBM Technical Disclosure Bulletin, "History Overflow Mechanism", L. Liu, J. Pomerene, T. R. Puzak, R. N. Rechtschaffen, P. Rosenfeld, F. Sparacio and J. Voldman, vol. 27, No. 5, pp. 3001–3002, Oct. 1984.

IBM Technical Disclosure Bulletin, "Adaptive Data Staging Mechanism in a Virtual Storage System", N. R. Clark and N. K. Ouchi, vol. 17, No. 1, pp. 210–214, Jun. 1974.

IBM Technical Disclosure Bulletin, "Simple Prefetching Scheme for Memory Hierarchy", K. So, vol. 26, No. 1, pp. 52–54, Jun. 1983.

Grimsrud, Knut S., "Multiple Prefetch Adaptive Disk Caching", IEEE Trans. on Knowledge and Data Eng, vol. 5, No. 1, Feb. 1993 pp. 88–103.

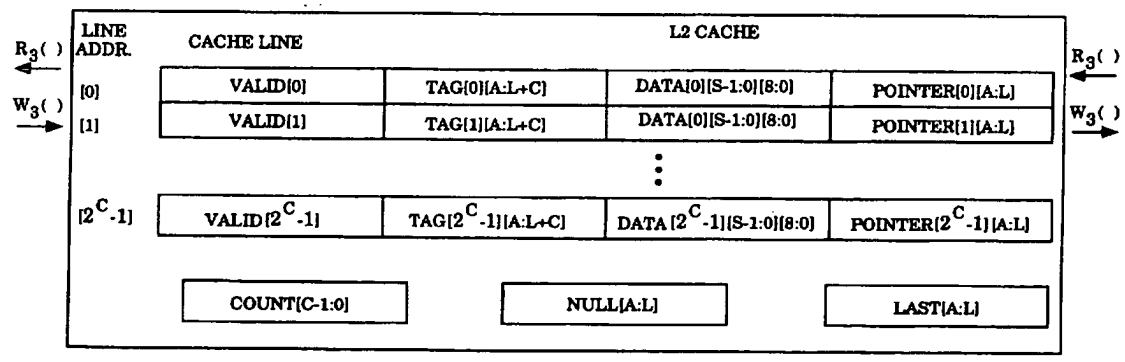
Primary Examiner—Rebecca L. Rudolph

Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor and Zafman

[57] **ABSTRACT**

A cache system which includes prefetch pointer fields for identifying lines of memory to prefetch thereby minimizing the occurrence of cache misses. This cache structure and method for implementing the same takes advantage of the previous execution history of the processor and the locality of reference exhibited by the requested addresses. In particular, each cache line contains a prefetch pointer field which contains a pointer to a line in memory to be prefetched and placed in the cache. By prefetching specified lines of data with temporal locality to the lines of data containing the prefetch pointers the number of cache misses is minimized.

80 Claims, 13 Drawing Sheets



US-PAT-NO: 5537573

DOCUMENT-IDENTIFIER: US 5537573 A

TITLE: Cache system and method for prefetching of data

DATE-ISSUED: July 16, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Ware; Frederick A.	Los Altos Hills	CA	N/A	N/A
Farmwald; Michael P.	Portola Valley	CA	N/A	N/A
Hampel; Craig	San Jose	CA	N/A	N/A
Krishnamohan; Karnamadakala	San Jose	CA	N/A	N/A

US-CL-CURRENT: 711/137, 711/213

ABSTRACT:

A cache system which includes prefetch pointer fields for identifying lines of memory to prefetch thereby minimizing the occurrence of cache misses. This cache structure and method for implementing the same takes advantage of the previous execution history of the processor and the locality of reference exhibited by the requested addresses. In particular, each cache line contains a prefetch pointer field which contains a pointer to a line in memory to be prefetched and placed in the cache. By prefetching specified lines of data with temporal locality to the lines of data containing the prefetch pointers the number of cache misses is minimized.

80 Claims, 15 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 13

---------- KWIC ---------

US Patent No. - PN (1):
5537573

Detailed Description Text - DETX (16):
Null[A:L]--a register which points to the last line accessed with a null pointer field.

Detailed Description Text - DETX (17):
Last[A:L]--a register which points to the last line accessed in the cache.

Detailed Description Text - DETX (20):
In the preferred embodiment, the pointer array is maintained in accordance with the access performed. Thus, if a cache miss occurs, the pointer field in a memory line which was previously accessed is changed so that a future miss at that address can be avoided. It is not necessary, however, that the pointer Y for a line of memory at address Y be in the line that was last accessed. As long as the pointer Y is placed in the pointer field of a line memory at address X which was accessed within the last 2.sup.C--1 requests, then it is probable that the line will still be in the cache when subsequently requested.

Detailed Description Text - DETX (24):
A generalized process for accessing the cache memory system will be described with reference to FIG. 8. At step 400, an index into the cache is generated. If, at step 410, the requested line is in the cache, at step 415 the data is returned to the processor, and at step 420, the pointer field of the line returned is temporarily saved. If the requested data is not located in the cache, the line is retrieved from main memory, step 430, and the pointer field is temporarily saved, step 435. At step 440, the previous line referenced which contained a null pointer or the last line referenced is identified. As a cache miss occurred because the line of data was not prefetched and placed in the cache, at step 445 a pointer of a recently accessed line is updated in main memory with a pointer identifying the line just referenced. In addition, the pointer of the line in the cache may also be updated. Preferably, a recently accessed line with a null pointer field is updated with the pointer to the referenced line. The line does not have to be

the last line accessed; so long as it is within the range corresponding to a predetermined number of lines of the cache, the likelihood that the line will be subsequently prefetched is increased. If a recently accessed line with a null pointer within the predetermined number of lines last accessed is not found, the last line previously accessed is updated with the pointer to the referenced line just accessed.

Detailed Description Text - DETX (32):

If one of the last 2.sup.C--1 lines supplied by the cache contains a Null value in its pointer field, then the line containing the Null pointer will store the address of this line. The address of that line is therefore written to the T2 temporary variable. The requested address will be written over the Null pointer value of that line. Preferably, value of all zeros will be used to identify a Null value, since a memory line should never need to prefetch itself. The Count counter is used to keep track of the number of lines that have been supplied by the cache since the last line with a Null value in the prefetch pointer field. If more than 2.sup.C--1 lines have been supplied, then the Last pointer will be used to identify the line in which the pointer is updated with the requested address. The Last pointer contains the address of the last line (before ReqAddr) supplied by the cache. The address of the last accessed line is therefore written into the T2 temporary variable. The requested address is written into the pointer field of the line in memory pointed to by the T2 temporary variable. The next time this earlier line is accessed by the cache, it will cause the current request address ReqAddr to be prefetched, thus avoiding another cache miss. This sequence establishes and maintains the pointer structure in main memory.

Claims Text - CLTX (67):

providing a Null pointer which points to the last recently accessed line in the main memory requested that contains a Null pointer value;